Project Number:	IST-4-027756
Project Title:	Wireless World Initiative New Radio – WINNER II
Document Type:	

Document Number:	
Document Title:	**Matlab SW documentation of WIM2 model**
Source Activity:	
Editor:	Pekka Kyösti
Authors:	Pekka Kyösti
Status / Issue:	0.4
Date Last changes:	23.9.2008
File:	WIM2-ITU Matlab documentation.doc

Abstract:	This is intended for the abstract

Keywords:

Document History:	
21.08.07 ver 0.1	(Pekka) Document created by modifying Phase I Matlab SW Documentation of the WIM model.doc (Authors: Jari Salo, Pekka Kyösti, Daniela Laselva, Giovanni DelGaldo)
9.11.07 ver 0.2	(Pekka) Document updated to Phase II final model
13.11.2007 ver 0.3	(Pekka) Input parameter tables refined.
23.9.2008 ver 0.4	(Pekka) Modified for IMT.EVAL channel model version

Table of Contents

1. Introduction and scope 3

2. Installation 3

3. Model features 4

3.1 Basic features 4
3.2 CDL model option 5
3.3 Supported propagation scenarios 5

4. Running examples 6

4.1 Basic examples of channel matrix generation 6

5. Technical description of the SW 6

5.1 SW high level structure 6
5.2 Model input/output interface 7
5.2.1 Input parameters 8
5.2.2 Output parameters 13
5.2.3 Scenario parameters 14
5.3 Implementation notes 14
5.3.1 Output power scaling 14
5.3.2 Output interpolation 14
5.3.3 Layout parameters (layoutparset.m) 14
5.3.4 System level simulations 17
5.4 ANSI-C speed-up implementation 17
5.4.1 A faster channel coefficients generation: scm_mex_core.m 17
5.4.2 A faster antenna pattern interpolation function: interp_gain_c.m 17

6. References 18

1. Introduction and scope

The channel modeling work package WP1 of IST-WINNER project provided a Matlab implementation of the WINNER Phase II Model (referred as WIM this onward). The main purpose of the WIM channel models is to generate a radio channel realisations for a link and system level simulations. The implementation is based on the earlier implementations of WINNER II interim model [6], WINNER I (interim) model [3], 3GPP/3GPP2 Spatial Channel Model (SCM) [4] and SCM(E) Extension models. For details concerning the WIM model, please refer to [1]. This document describes the SW implementation structure of WIM, its input/output interface and its functionality. Channel model scenarios and parameters are modified to correspond to IMT.EVAL [7] channel model.

The channel model takes the user defined parameters, the MIMO radio link parameters and antenna parameters described in [1] as an input. Channel matrices can be generated for multiple BS-MS links with one function call. The output is a multi-dimensional array which contains the channel impulse responses for the given radio links. In addition, the randomly drawn channel parameters for each link will be given as an output.

The channel convolution and other related operations are beyond the scope of the implemented channel model.

2. Installation

The WIM package installs as a MATLAB mini-toolbox. Unzip the files in their own directory, e.g. ‘winner‘. Add the directory to MATLAB path. Type ‘help winner’ at MATLAB command to get started.

The package includes the following modules:

```
% IMT.EVAL channel model
%   based on Winner Phase II channel model
% Version 0.2, September 22, 2008
%
% Channel model functions
%   wim               - WINNER Phase II channel model (D1.1.2)
%   wimparset         - Model parameter configuration for WIM
%   linkparset        - Link parameter configuration for WIM
%   layoutparset      - Layout parameter configuration for WIM (optional)
%   antparset         - Antenna parameter configuration for WIM
%   pathloss          - Pathloss models for 2GHz and 5GHz
%
% WINNER -specific functions
%   scenpartables     - Set WIM parameters for WINNER scenarios
%
% Miscellaneous functions
%   cas               - Circular angle spread (3GPP TR 25.996)
%   ds                - RMS delay spread
%   dipole            - Field pattern of half wavelength dipole
%   NTlayout          - Visualisation of network layout
%
% Utility functions
%   interp_gain       - Antenna field pattern interpolation
%   interp_gain_c     - Antenna field pattern interpolation (requires GSL)
%   wim_core          - Channel coefficient computation for a geometric channel model
%   scm_mex_core      - WIM_CORE written in ANSI-C
%   generate_bulk_par - Generation of WIM bulk parameters
%   layout2link       - Computes and converts layout to link parameters
%   ScenarioMapping   - Maps scenario names (A1 etc.) to number indices
%   struct_generation - Assistant function
%   offset_matrix_generation - Assistant function
```

Note! The code is created and tested on Matlab versions 6.5 and 7.1 (R14). Some other version might cause problems.
To compile the ANSI-C functions (optional) MATLAB’s mex compiler must be properly configured.

3. Model features

3.1 Basic features

WINNER MIMO radio channel model enables system level simulations and testing. This means that multiple links are to be simulated (evolved) simultaneously. System level simulation may include multiple base stations, multiple relay stations, and multiple mobile terminals as in Figure 1. Link level simulation is done for one link, which is shown by blue dashed ellipse. The short blue lines represent channel segments where large scale parameters are fixed. System level simulation consists of multiple links. Both link level and system level simulations can be done by modelling multiple segments, or by only one (CDL model).

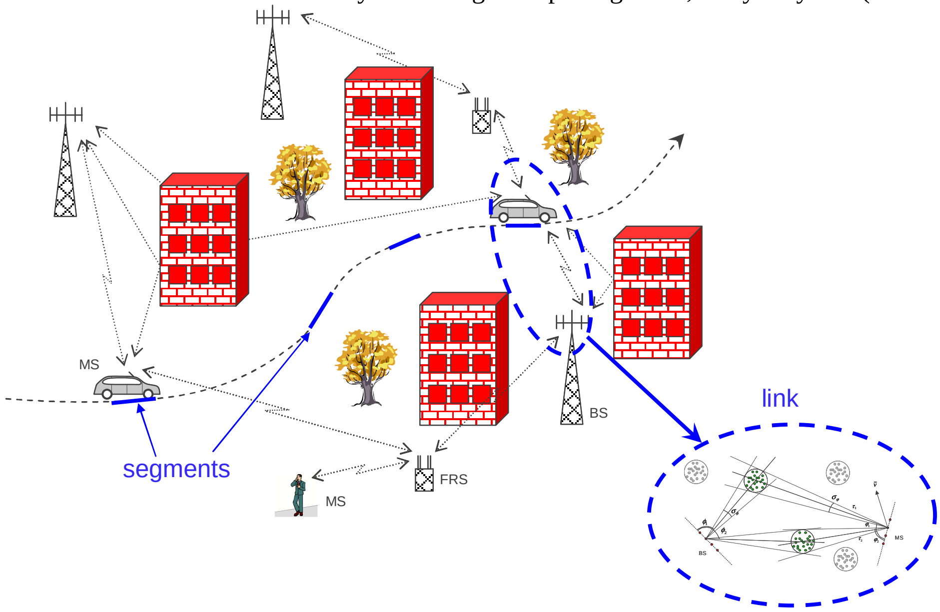

Figure 1. System level approach, several segments (drops).

A single link model is shown in Figure 2. The parameters used in the models are also shown in the figure. Each circle with several dots represents scattering region causing one cluster. The number of clusters varies from scenario to another.

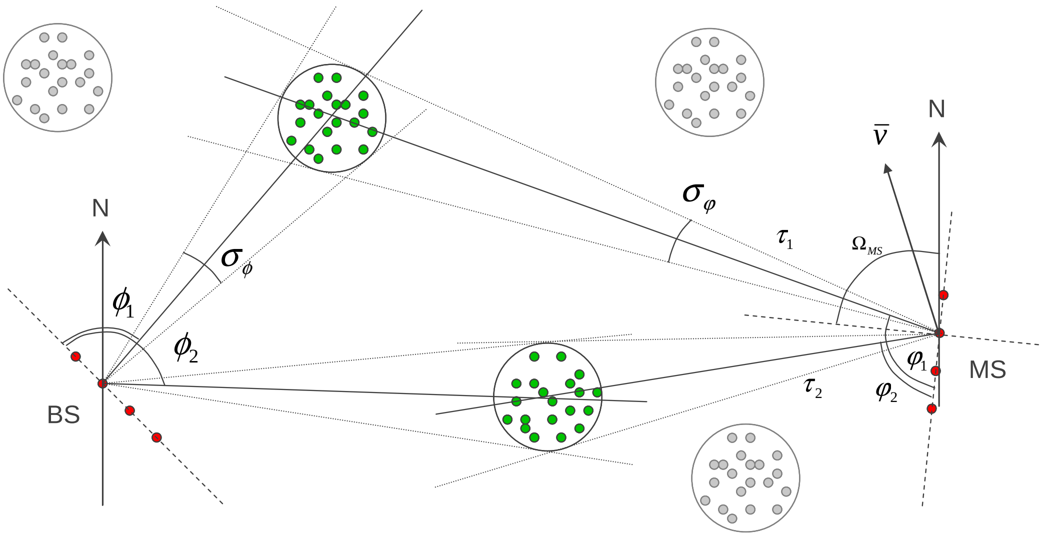

Figure 2. Single link.

By selecting the WIM input parameters suitably, it is possible to generate with a single function call:

- Multiple channel realizations for a single BS-MS link,
- One channel realization for multiple BS-MS links,
- Or both.

E.g.,
```
% to generate matrices for 10 links with default parameters, default scenario InH (A2)
H=wim(wimparset,linkparset(10),antparset);

% to generate a single link of any of the supported scenarios (Scenario 1=InH (A2), 2=UMi
(B1), 3=SMa (C1), 4=UMa (C2), 5=UMi O-2-I (B4) 6=RMa (D1))
linkpar=linkparset(1); linkpar.ScenarioVector=3;     % E.g. SMa
H=wim(wimparset,linkpar,antparset);
```

The number of paths is scenario dependent and varies from 8 to 24 (see [1]). The number of rays (subpaths) is fixed to 20 for all scenarios.

3.2 CDL model option

IMT.EVAL [7] contains fixed channel models called Clustered Delay Line (CDL) models with tabulated spatio-temporal channel parameters. They are fully deterministic despite random initial phases of the rays. Doppler is not explicitly defined, because it results from the angular properties. These models might be useful e.g. in the simulation system calibration simulations. With CDL models there is no randomness in the large scale parameters. **NOTE! In the current version CDL models are not supported.**

CDL models can be selected by setting parameters:
`wimpar.FixedPdpUsed ='yes'` and `wimpar.FixedAnglesUsed ='yes'`.

3.3 Supported propagation scenarios

For details about the scenarios definition see [7]. Spatial channel model parameters characterize each channel scenario. The scenarios-dependent parameters are currently supported at center frequency of 2-6 GHz and at bandwidth of 100 MHz.

Test environment	Base coverage urban	Microcellular	Indoor	High speed
Deployment scenario	Urban macro-cell scenario	Urban micro-cell scenario	Indoor hotspot scenario	Rural macro-cell scenario
Channel model	UMa Urban macro (LOS, NLOS)	UMi Urban micro (LOS, NLOS, Outdoor-to-indoor)	InH Indoor hotspot (LOS, NLOS)	RMa Rural macro (LOS, NLOS)
Acronym in implementation	C2	B1 (LOS&NLOS) B4 (Out-to-in)	A2	D1

4. Running examples

4.1 Basic examples of channel matrix generation

```
%% Matrix generation for 10 MS-BS links with default settings
H=wim(wimparset,linkparset(10),antparset);

%% Matrix generation with modified parameter settings
% Setting and modifying default input parameters
wimpar=wimparset;
linkpar=linkparset(10);  % 10 links
antpar=antparset;        % default antennas
wimpar.NumTimeSamples=100;      % 100 time samples per link
wimpar.NumBsElements = 4;       % set 4 BS elements
wimpar.NumMsElements = 4;       % set 4 MS elements
antpar.BsElementPosition = 10;  % set 10 lambda spacing
antpar.MsElementPosition = 0.5; % set 0.5 lambda spacing
linkpar.PropagConditionVector=zeros(1,10);  % (NLOS=0/LOS=1)
linkpar.ScenarioVector=4*ones(1,10);  % UMa scenario
% Generate channel realisations
[H1,delays,out]=wim(wimpar,linkpar,antpar);
% using final conditions as initial conditions in next function call
[H2,delays,out]=wim(wimpar,linkpar,antpar,out);
```

5. Technical description of the SW

5.1 SW high level structure

The high-level WIM implementation structure is shown in the block diagram given in Figure 3. It is assumed that the user mobility model, which is not specified in [1] is external to the channel matrix generation routine. The path loss model is also implemented as a separate user-supplied function. The default path loss function, complying with [1], is [pathloss.m]. Interpolation of antenna field patterns is also required since AoD/AoAs can be any values over (-180,180) degrees. This function is also implemented as an external function, so that model user can supply own function, if desired.

The WIM computation consists of two main parts:

- the random user parameter generation and
- the actual channel matrix computation.

Input and output arguments are defined in more detail in the next section.

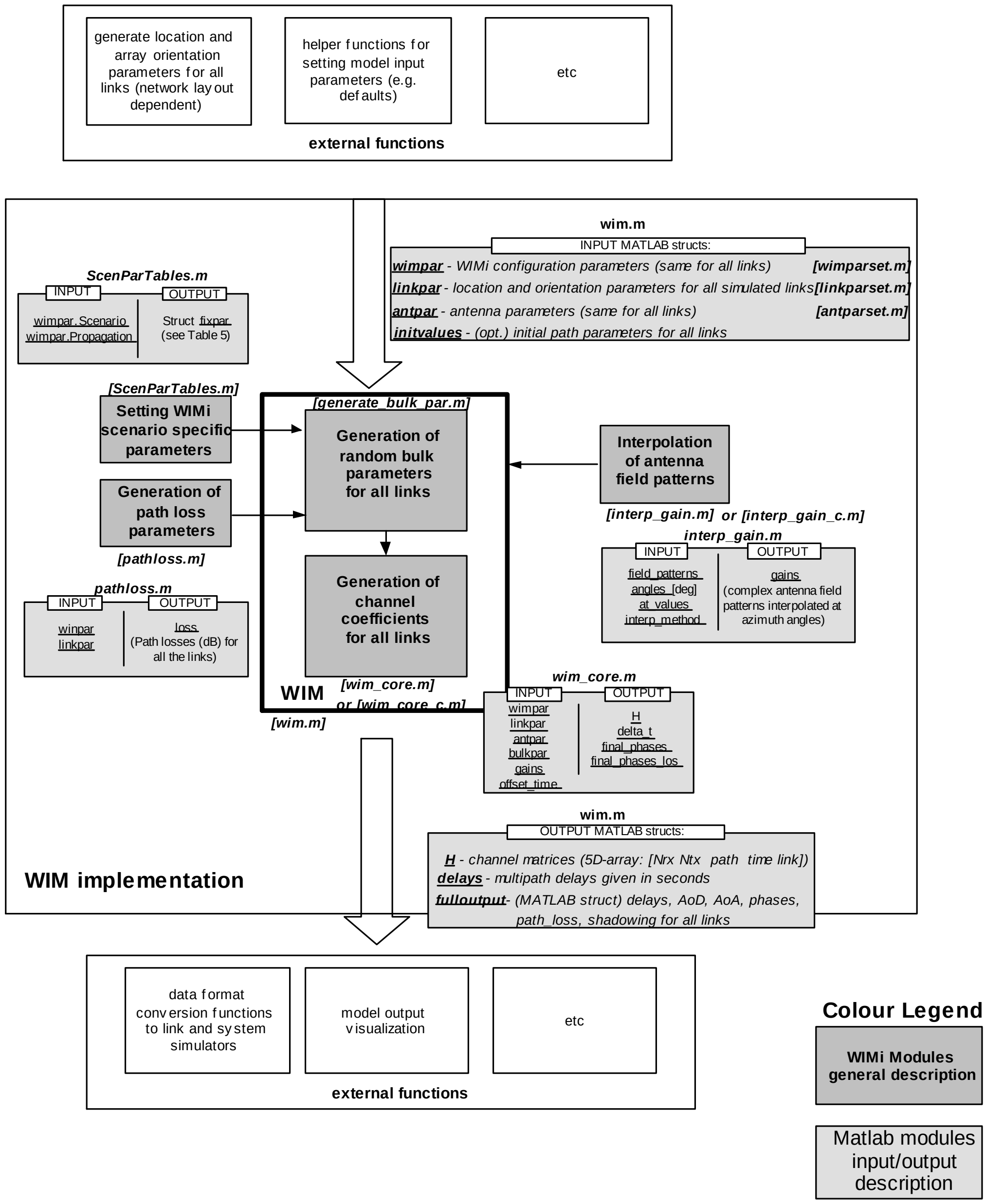

Figure 4. High-level description of the WIM computation. The actual WIM model is in the box labeled 'WIM'.

5.2 Model input/output interface

The full syntax for the WIM function is ([.] indicate optional arguments):

[H, [DELAYS], [FULL_OUTPUT]] = WIM(WIMPAR, LINKPAR, ANTPAR, [INITVALUES]).

- All input arguments are MATLAB structs. The first three input arguments are mandatory. A helper functions (wimparset.m, linkparset.m and antparset.m) will be supplied so that their default values can be set easily.
- The fourth input argument is optional. When given, WIM does not generate the channel parameters randomly, but uses the supplied initial channel values.
- The first output argument is a 5D-array containing the MIMO channel matrices for all links over a specified number of time samples.
- The second output argument includes multipath delays for all links, given in [s].
- The third output argument is a MATLAB struct containing the randomly generated link parameters and the final phases of the complex sinusoids. This MATLAB struct can be used as INITVALUES in subsequent function calls to generate time continuous channel realizations with separate function calls.

5.2.1 Input parameters

Tables 1-4 describe the fields of the input structs WIMPAR, LINKPAR, and ANTPAR and Table 5 includes the fixed input parameters.

Table 1: MATLAB struct WIMPAR (set in wimparset.m). General channel model parameters, common for all links.

Parameter name	Definition	Default value	Unit	Note
NumBsElements	Number of BS array antenna elements. The number of BS and MS elements is normally extracted from ANTPAR. The values of NumBsElements and NumMsElements are used only if a single scalar is given as the antenna field pattern in ANTPAR (see ANTPARSET).	2	-	Ignored if antenna patterns are defined in input struct ANTPAR, and number of BS elements is extracted from the antenna definition.
NumMsElements	Number of MS array antenna elements. The number of BS and MS elements is normally extracted from ANTPAR. The values of NumBsElements and NumMsElements are used only if a single scalar is given as the antenna field pattern in ANTPAR (see ANTPARSET).	2	-	Ignored if antenna patterns are defined in input struct ANTPAR, and number of MS elements is extracted from the antenna definition.
SampleDensity	Oversampling factor, number of time samples per half wavelength. For successful Doppler analysis, one should select SampleDensity > 1. The time sample interval is calculated from CenterFrequency and MsVelocity (see LINKPARSET) according to wavelength/(MsVelocity*SampleDensity). The calculated time sample interval for each link is included in the optional output argument of WIM.	2	-	
NumTimeSamples	Number of time samples	100	-	
UniformTimeSampling	If UniformTimeSampling is 'yes' all links will be sampled at simultaneous time instants. In this case, the time sample interval is the same for all links it is calculated by replacing MsVelocity with MAX(MsVelocity), where the maximum is over all links. If 'no' all the links are time sampled with different rate depending on MsVelocity.	'no'	-	'Yes' setting could be useful in some system-level simulations where all simulated links need to be sampled at equal time intervals, regardless of MS speeds.
IntraClusterDsUsed	If 'yes' the two strongest clusters in power are divided in delay into three subclusters. Fixed delays are [0 5 10] ns, fixed powers are [10 6 4]/20. For details see [1, section 4.2]. Number of delay tap grows by four. If 'no', the clusters are not spread in delay.	'yes'	-	
NumSubPathsPerPath	Number of rays (i.e. complex sinusoids, plane waves) per cluster. It is not possible to change this value from 20 without modifying the code.	20	-	This is a fixed value.
FixedPdpUsed	If 'yes' the power and delay parameters are not drawn randomly, but taken from the CDL parameter tables [1, table 6-1..26]. In the default mode 'no', the parameters are random variables.	'no'	-	

FixedAnglesUsed	If 'yes' the angle parameters are not drawn randomly, but taken from the CDL parameter tables [1, table 6-1..26] and the random pairing of AoDs and AoAs is not used. In the default mode 'no', the parameters are random variables.	'no'	-	
PolarisedArrays	If PolarisedArrays='yes', single channel coefficient of impulse response turns to 2x2 coefficient matrix, with elements [VV VH;HV HH]. Where V stands for vertical polarisation and H for horizontal.	'no'	-	
CenterFrequency	The carrier center frequency. Center frequency affects path loss and time sampling interval.	5.25e9	Hz	
DelaySamplingInterval	DelaySamplingInterval determines the sampling grid in delay domain. All path delays are rounded to the nearest grid point. It can also be set to zero.	5e-9	sec	
PathLossModelUsed	When PathLossModelUsed is 'no' the path losses are still computed for each link but they are not multiplied into the channel matrices. Path loss is given only as an output parameter. If 'yes', path loss is multiplied to channel matrices.	'no'	-	
ShadowingModelUsed	When ShadowingModelUsed is 'no' the shadowing coefficients are still computed for each link but they are not multiplied into the channel matrices. Shadowing is given only as an output parameter. If 'yes', shadowing is multiplied to channel matrices.	'no'	-	
PathLossModel	The path loss model function name. Path loss model is implemented in a separate function, whose name is defined in PathLossModel. For syntax, see PATHLOSS. The default function is PATHLOSS, which complies with [1].	'pathloss'	-	
AnsiC_core	Use optimized computation. The C-function must be compiled before usage. For more information of the ANSI-C core function, see SCM_MEX_CORE.	'no'	-	Not supported with many options.
LookUpTable	The LookUpTable parameter defines the number of points used in the cosine look-up table; a power-of-2 should be given. The look-up table is used only in the ANSI-C optimized core function. Value 0 indicates that look-up table is not used. Value -1 uses the default number of points, which is 2^14=16384. Since a large part of computation in WIM involves repeated evaluation of a complex exponential, the look-up table can speed up computation on certain platforms and C compilers.	0	-	Applied only with AnsiC core.
RandomSeed	Sets random seed for Matlab random number generators. The default value is empty. Even fixing the random seed may not result in fully repeatable simulations due to differences in e.g. MATLAB versions.	[ ]	-	see Matlab help
UseManualPropCondition	If 'yes' the propagation condition (los/nlos) setting is defined manually in LINKPARSET. If 'no', the propagation condition is drawn from LOS probabilities in [1, table 4-7].	'yes'	-	
range	Path loss parameter for scenario B5b. In B5b the path-loss ranges 1, 2 and 3 are defined, see [1 table 6-20].	1	-	
end_time	Observation end time for B5 scenarios time points are taken as: wimpar.TimeVector=linspace(0,wimpar.end_time,T);	1	sec	

Table 2: MATLAB struct <u>LINKPAR</u> (set in linkparset.m). Link-dependent length k vectors, randomly generated; they are not based on any specific network geometry or user behaviour model.

Parameter name	Definition	Default value	Unit	Note
ScenarioVector	A 1xK vector mapping scenarios to links. Scenarios are [1=A1, 2=A2, 3=B1, 4=B2, 5=B3, 6=B4, 7=B5a, 8=B5c, 9=B5f, 10=C1, 11=C2, 12=C3, 13=C4, 14=D1, 15=D2a].	ones(1,K)	{1,2, …,15}	
PropagConditionVector	A 1xK vector mapping propagation condition (NLOS/LOS) to links. If WIMPAR UseManualPropCondition = 'yes', link propagation conditions (NLOS=0/LOS=1) are defined by this vector.	zeros(1,K)	{0,1}	Possible values 0=NLOS and 1=LOS.

MsBsDistance	A 1xK vector defining distance between BS and MS for K links. It is recommended to define MsBsDistance manually. **If** random MsBsDistance is used, **set feasible** values for RMIN and RMAX (see LINKPARSET). This is not done automatically for different scenarios.	*see LINKPARSET*	m	
BsHeight	A 1xK vector defining BS height from ground level. If NaN the height values are scenario dependent default heights from [1, table 4-4].	NaN	m	
MsHeight	A 1xK vector defining MS height from ground level. If NaN the height values are scenario dependent default heights from [1, table 4-4].	NaN	m	
ThetaBs	θ_{BS} (see Figure 5)	360*(rand(1,K)-0.5)	deg	U(-180,180)
ThetaMs	θ_{MS} (see Figure 5)	360*rand(1,K)-0.5)	deg	U(-180,180)
MsVelocity	MS velocity		m/s	
MsDirection	θ_v (see Figure 5)	360*(rand(1,K)-0.5)	deg	U(-180,180)
StreetWidth	A parameter for B1 and B2 path loss model. Average width of the streets, same for all users.	20	m	
NumFloors	A parameter for A2/B4 path loss model. NumFloor is the floor number in which the indoor MS/BS is located. E.g. in A2 scenario NumFloors is 5 if BS is located on the 5th floor. On ground floor (=street level) NumFloor = 0.	1	-	
NumPenetratedFloors	A parameter for A1 NLOS path loss model [1, table 4-4]. Number of penetrated floors between BS and MS.	0	-	
LayoutType	Layout type for UMi (B1/B4) path loss, '0'=hexagonal, '1'=Manhattan. This parameter is necessary only for UMi path loss function.	0	-	
OtoI_OutdoorPL	Outdoor-to-Indoor propagation condition for UMi (see. note 3 in A in A1-2, [7]). '1' denotes LOS, '0' denotes NLOS. This parameter is necessary only for UMi O-to-I path loss function.	1	-	Possible values 0=NLOS and 1=LOS.
Dist1	Distance definition for B1 and B2 path loss model. Dist1 is a distance from BS to the "last line-of-sight point", typically street crossing, see [1, fig 4-3]. Default value is NaN, which denotes random distance determination in PATHLOSS function.	NaN	-	
BuildingHeight	Average building height, same for all users. Necessary only for path loss models. If not set by the user (i.e. NaN) the path loss function will use default scenario dependent heights.	NaN	m	

Table 3. MATLAB struct LAYOUTPAR (set in layoutparset.m): network layout parameters.

Parameter name	Definition	Default value	Unit	Note
BsXY	A 2xNofBs matrix of BS (x,y) co-ordinates. Co-ordinates of Bs and Ms should be given in meters with resolution of 1 meter. It is recommended to define BsXY manually. **If** random BsXY is used, **set a feasible** value for RMAX (see LAYOUTPARSET). As a default x,y co-ordinates are drawn uniformly to RMAX x RMAX square (RMAX=500).	round(rand(2,NofBs)*rmax)	m	
NofSect	A parameter defining the number of sectors in each of the BSs. Can be either a scalar or a vector. If scalar all the BS has the same number of sectors.	ones(1, NofBs)	-	
BsOmega	BsOmega is a matrix with dimensions max(NofSect)xNofBs. Each column of the matrix contains orientations of sectorised arrays with respect to some fixed North direction. If some BS have less sectors than others, the non-existing sector orientations are set to zero. E.g.	*see LAYOUTPARSET*	deg	

	setup with 2 BS, one with 1 sector and other with 3 sectors. In this case orientation matrix could be e.g. BsOmega = [11 22; 0 33; 0 44]. See [1, Fig 5-2].			
MsXY	A 2xNofMs matrix of MS (x,y) co-ordinates. Co-ordinates of Ms should be given in meters with resolution of 1 meter. It is recommended to define BsXY manually. **If** random MsXY is used, **set a feasible** value for RMAX (see LAYOUTPARSET). As a default x,y co-ordinates are drawn uniformly to RMAX x RMAX square (RMAX=500).	round(rand(2,NofMs)*rmax)	m	
MsOmega	A 1xK vector of MS array broad side orientations. See [1, Fig 5-2]	360*(rand(1,NofMs)-0.5)	deg	
Pairing	Pairing is a matrix with dimensions NofSect x NofMs, i.e. one entry for each BS sector/MS pair. Value '1' stands for "link will be modelled" and value '0' stands for "link will not be modelled". E.g. with all ones matrix, all the MS are connected to all sectors. With e.g. first rows ones and others zeros means, that all MS are connected to only 1st sector of 1st BS. See [1, sect 5.1.1].	*see LAYOUTPARSET*	-	
ScenarioVector	Same as in LINKPAR (see Table 2).			
PropagConditionVector				
BsHeight				
MsHeight				
MsVelocity				
MsDirection				
StreetWidth				
NumFloors				
NumPenetratedFloors				
Dist1				

Table 4. MATLAB struct ANTPAR (set in antparset.m): antenna parameters. Linear uniform arrays with are supported.

The antenna patterns do not have to be identical. The complex field pattern values for the randomly generated AoDs and AoAs are interpolated.

Parameter name	Definition	Default value	Unit	Note
BsGainPattern	BS antenna field pattern values in a 4D array. The dimensions are [ELNUM POL EL AZ] = SIZE(BsGainPattern), where ELNUM - the number of physical antenna elements in the array. The elements may be dual-polarized. POL – polarization. The first dimension is vertical polarization, the second is horizontal. If the polarization option is not used, vertical polarization is assumed (if both are given). EL – elevation. This value is ignored. Only the first element of this dimension is used. AZ – complex-valued field pattern in the azimuth dimension given at azimuth angles defined in BsGainAnglesAz. If NUMEL(BsGainPattern)=1, all elements are assumed to have uniform gain defined by the value of BsGainPattern over the full azimuth angle, and the number of BS antenna elements is defined by NumBsElements. This speeds up computation since field pattern interpolation is not required.	1	-	
BsGainAnglesAz	Vector containing the azimuth angles for the BS antenna field pattern values. These values are assumed to be the same for both polarizations. This value is given in degrees	linspace(-180,180,90)	deg	

	over the range (-180,180) degrees. If NUMEL(BsGainPattern)=1, this variable is ignored.			
BsGainAnglesEl	Vector of elevation angles for definition of BS antenna gain values. This parameter is for future needs only; its value is ignored in this implementation (SCM does not support elevation).	-	-	
BsElementPosition	Element spacing for BS linear array in wavelengths. This parameter can be either scalar or vector. If scalar, uniform spacing is applied. If vector, values give distances between adjacent elements.	0.5	wavelength	
MsGainPattern	MS antenna field pattern values in a 4D array. The dimensions are [ELNUM POL EL AZ] = SIZE(MsGainPattern), where ELNUM – the number of physical antenna elements in the array. The elements may be dual-polarized. POL – polarization. The first dimension is vertical polarization, the second is horizontal. If the polarization option is not used, vertical polarization is assumed (if both are given). EL – elevation. This value is ignored. Only the first element of this dimension is used. AZ – complex-valued field pattern in the azimuth dimension given at azimuth angles defined in MsGainAnglesAz. If NUMEL(MsGainPattern)=1, all elements are assumed to have uniform gain defined by the value of MsGainPattern over the full azimuth angle, and the number of MS antenna elements is defined by scmpar.NumMsElements. This speeds up computation since field pattern interpolation is not needed.	1	complex	
MsGainAnglesAz	Vector containing the azimuth angles for the MS antenna field pattern values. These values are assumed to be the same for both polarizations. This value is given in degrees over the range (-180,180) degrees. If NUMEL(BsGainPattern)=1, this variable is ignored.	linspace(-180,180,90)	deg	
MsGainAnglesEl	Vector of elevation angles for definition of MS antenna gain values. This parameter is for future needs only; its value is ignored in this implementation (SCM does not support elevation).	-	-	
MsElementPosition	Element spacing for MS linear array in wavelengths. This parameter can be either scalar or vector. If scalar, uniform spacing is applied. If vector, values give distances between adjacent elements.	0.5	wavelength	
InterpFunction	The name of the interpolating function. One can replace this with his own function. For syntax, see interp_gain.m, which is the default function. For faster computation, see interp_gain_c.m	‘interp_gain’		
InterpMethod	The interpolation method used by the interpolating function. Available methods depend on the function. The default function is based on MATLAB’s interp1.m function and supports e.g. ‘linear’ and ‘cubic’ (default) methods. Note that some methods, such as ‘linear’, cannot extrapolate values falling outside the field pattern definition.	‘cubic’		

Note that the mean power of narrowband channel matrix elements (i.e. summed over delay domain) depends on the antenna gains

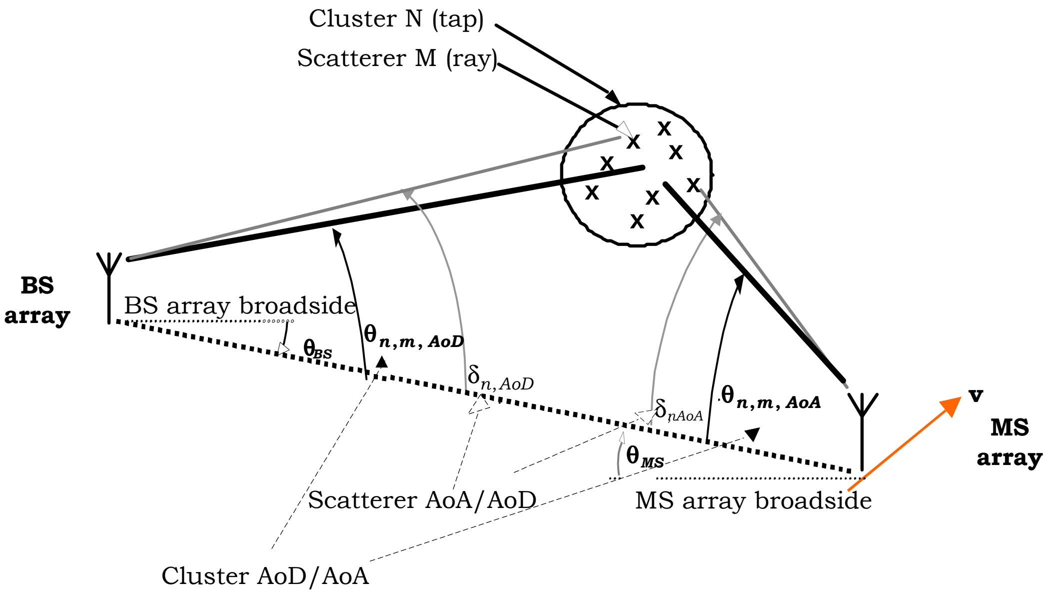

Figure 5: BS and MS angular parameters in WIM.

The fourth input argument, which is also a MATLAB struct, is optional. It can be used to specify the initial AoDs, AoAs, cisoid phases, path losses and shadowing values when WIM is called recursively, or for testing purposes. If this argument is given, the random parameter generation as defined in [1] is not needed. Only the antenna gain values will be interpolated for the supplied AoAs and AoDs.
The fields of the MATLAB struct are given in the following table. Note! The fourth input argument can be directly the output structure FULLOUTPUT defined in Table 6 or the structure defined in Table 5.

Table 5. MATLAB struct INITVALUES: initial values, fourth optional input argument.

Parameter name	**Definition**	**Unit**
InitDelays	A K x N matrix of path delays.	Sec
InitSubPathPowers	A K x N x M array of powers of the subpaths.	-
InitAods	A K x N x M array	Degrees
InitAoas	A K x N x M array	Degrees
InitSubPathPhases	A complex-valued K x N x M array. When polarization option is used, this is a K x P x N x M array, where P=4. In this case the second dimension includes the phases for [VV VH HV HH] polarized components.	Degrees
InitPathLosses	A K x 1 vector	Decibel
InitShadowLosses	A K x 1 vector	Decibel
InitIndOfSpreadClust	A K x 2 matrix. Index to two strongest clusters. These clusters are spread to three delay positions if parameter IntraClusterDsUsed = 'yes'	

5.2.2 Output parameters

There are three output arguments: H, DELAYS, FULLOUTPUT. The last two are optional.

Table 6. MATLAB output parameters.

Parameter name	**Definition**	**Unit**	**Note**
H	A 5D-array with dimensions U x S x N x T x K		See Appendix 1 for terminology
DELAYS	A K x N vector of path delay values. Note that delays are, for compatibility with the INITVALUES, also included in FULLOUTPUT..	sec	
FULLOUTPUT	A MATLAB struct with the following elements:		

delays	A K x N matrix of path delays. This is identical to the second output argument.	sec	
path_powers	A K x N array of path powers.	linear	
aods	A K x N x M array of subpath angles of departure	degrees	
aoas	A K x N x M array of subpath angles of arrival	degrees	
path_losses	A K x 1 vector	linear scale	
MsBsDistance	1 x K vector of MS-BS distances	m	
shadow_fading	A K x 1 vector	linear scale	
sigmas	A K x 4 vector of per link large scale parameters (ASD ASA DS SF)	-	
propag_condition	A K x 1 vector indicating LOS / NLOS condition (0=NLOS, 1=LOS)	-	
Kcluster	A K x 1 vector defining narrowband K-factors of links.	linear scale	Only with LOS
Phi_LOS	Final phases for LOS paths, K x 1 array.	deg	Only with LOS
scatterer_freq	A K x N x M array of subpath Doppler frequecies	Hz	Only with B5 CDL
subpath_phases	A complex-valued K x N x M array giving the final phases of all subpaths. When polarization option is used, a K x P x N x M array, where P=4. In this case the second dimension includes the phases for [VV VH HV HH] polarized components.	degrees	
delta_t	A K x 1 vector defining time sampling interval for all links.	sec	
IndexOfDividedClust	A K x 2 matrix. Index to two strongest clusters. These clusters are spread to three delay positions if parameter IntraClusterDsUsed = ‘yes’	-	
xpr	A K x N x M array of cross-polarization coupling power ratios.	linear scale	Only with PolarisedArrays case.

5.2.3 Scenario parameters

In addition, fixed parameters are provided in the function [ScenParTables.m].

5.3 Implementation notes

5.3.1 Output power scaling

In the basic case of NLOS propagation condition with non-polarised, isotropic antenna patterns, the mean output power of each MIMO channel is unity (without shadowing and pathloss). In the cases of LOS condition or non-isotropic antenna patterns or with with dual polarized arrays, mean power varies.

5.3.2 Output interpolation

Channel sampling frequency has to be finally equal to the simulation system sampling frequency. To have feasible computational complexity it is not possible to generate channel realisations on the sampling frequency of the system to be simulated. The channel realisations have to be generated on some lower sampling frequency and then interpolated to the desired frequency. A practical solution is e.g. to generate channel samples with sample density (over-sampling factor) two, interpolate them accurately to sample density 64 and to apply zero order hold interpolation to the system sampling frequency. Channel impulse responses can be generated during the simulation or stored on a file before the simulation on low sample density. Interpolation can be done during the system simulation.

5.3.3 Layout parameters (layoutparset.m)

This implementation supports a multi-base station and multi-mobile station network layout. As an option it is possible to define the layout parameters also in the Cartesian coordinate system. In such a case the network layout includes information about: the numbers and locations of MSs and BSs in the Cartesian coordinates; the number of sectors in a BS (in case of a multi-cell network); the array broad side orientations at both MS and BS; the coupling of an active radio link from a MS to a certain sector of a BS (or vice-versa); and the directions of the MSs.

The utility function layoutparset.m is used to generate link parameters the following way:

```
>> linkpar = layout2link ( layoutparset ( NofMs, NofBs, SectPerBs, K ) )
% SectPerBs = # sectors in a BS
% K= # of active links
```

It is also possible and recommended to define the network layout manually by directly editing all or part of the layout parameters in the `layout` structure.

Within e.g. the $100x100m^2$ cell area MSs and BSs locations and radio active links are randomly generated. A pairing matrix selects which radio links will be generated (active) and which will not The pairing matrix is a [NofMs x (NofSect x NofBs)] matrix with values {0,1}. Value 0 stands for the link MS*m* to cell*n* (sector*n)* that is not modelled and value 1 for the link if modelled. The pairing matrix **A** in the example below is a 3x6 matrix, in the case 3 MSs, 2 BSs and 3 sectors or 3 MSs, 6 BSs and 1 sector.

$$\mathbf{A} = \begin{bmatrix} \chi_{c1,ms1} & \chi_{c1,ms2} & \chi_{c1,ms3} \\ \chi_{c2,ms1} & \chi_{c2,ms2} & \chi_{c2,ms3} \\ \vdots & \vdots & \vdots \\ \chi_{c6,ms1} & \chi_{c6,ms2} & \chi_{c6,ms3} \end{bmatrix}$$

Both the distance and line of sight (LOS) direction information of the radio links are calculated for the input of the model. The distance between the BS_i and MS_k is

$$d_{BS_i,MS_k} = \sqrt{(x_{BS_i} - x_{MS_k})^2 + (y_{BS_i} - y_{MS_k})^2}.$$

The LOS direction from BS_i to MS_k with respect to BS antenna array broad side is (see Figure 6)

$$\theta_{BS_i,MS_k} = \begin{cases} -\arctan\left(\dfrac{y_{MS_k} - y_{BS_i}}{x_{MS_k} - x_{BS_i}}\right) + 90^\circ - \Omega_{BS_i}, & \text{when } x_{MS_k} \geq x_{BS_i} \\ -\arctan\left(\dfrac{y_{MS_k} - y_{BS_i}}{x_{MS_k} - x_{BS_i}}\right) - 90^\circ - \Omega_{BS_i}, & \text{when } x_{MS_k} < x_{BS_i} \end{cases}$$

The angles and orientations are depicted in the figure below. The north (up) is the zero angle reference and the positive direction of the angles is the clockwise direction.

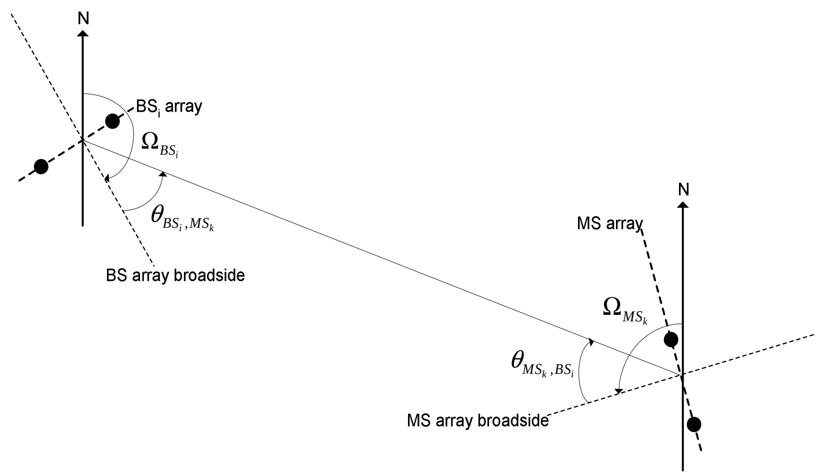

Figure 6: BS and MS antenna array orientations.

A rudimental visualization of the network layout is implemented as helper utility in order to visualize the layout of the generated channel; the network layout with 5 BSs and MSs, and 7 active links looks like as in the following figures. In the figures red arrows denote the BS sector array orientation, black arrows denote the MS direction of motion, and blue arrows denote the active (modelled) links.

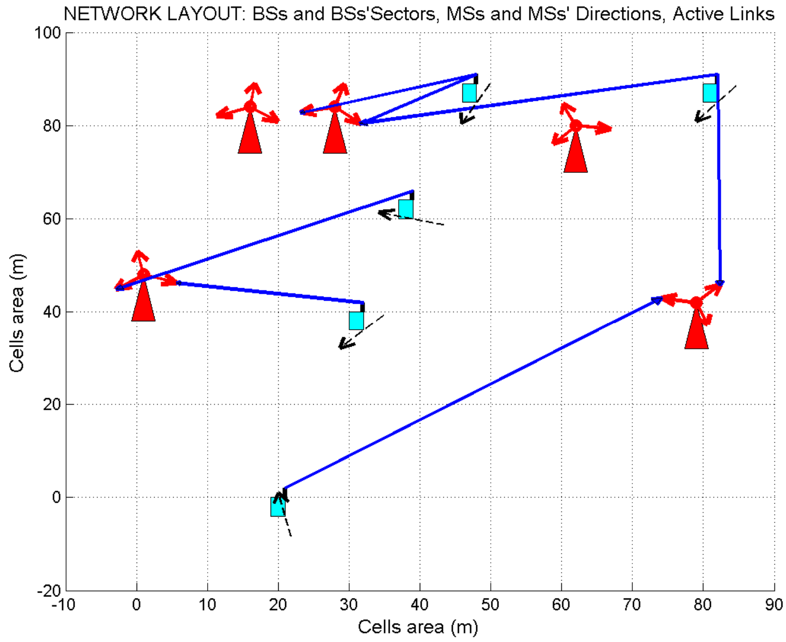

Figure 7: Example of the network layout with 5 BSs (three sectors in any BS), 5 MSs and 7 active links

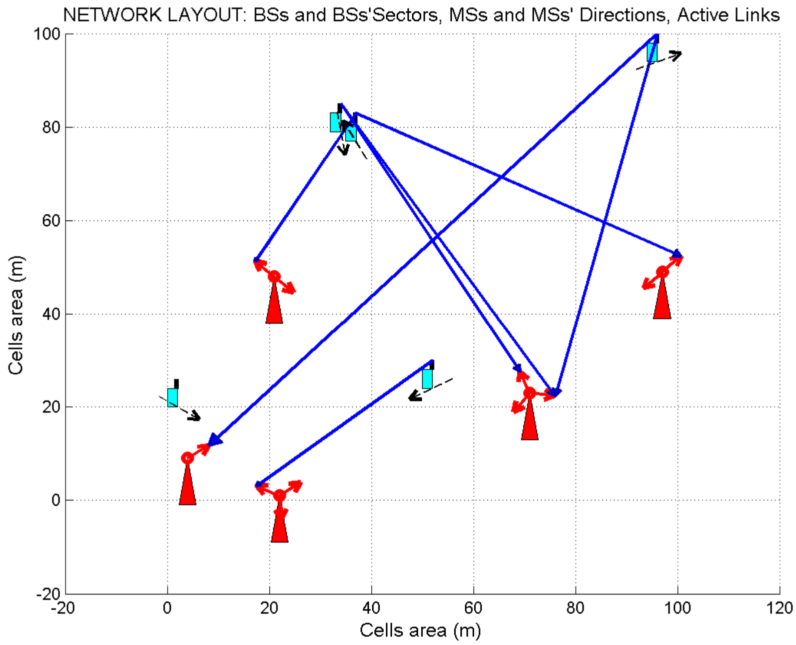

Figure 8: Example of the network layout with 5 BSs (number of sectors varies from 1-3 in different BSs), 5 MSs and 7 active links

The utility function NTlayout.m is used the following way:

```
>> NTlayout ( layout2link ( layoutparset ( NoMs, NofBs, SectPerBs, K ) ) )
% SectPerBs = # sectors in a BS
% K= # of active links
```

By setting the MSs and BSs coordinates and selecting the Pairing matrix properly it is possible to simulate many system-level cases (see next paragraph).

5.3.4 System level simulations

Guidelines for system levels simulations are in [1]. There are described cases like handover, multi-cell, multi-user, multi-hop and relaying simulations.

NOTE! If a parameter MsBsDistance is not defined by the user, it will be drawn randomly in function [linkparset.m]. Random distances are drawn such that the MSs would be approximately uniformly distributed in a circular disk over [RMIN,RMAX] meters. Default values for RMIN and RMAX are 5 and 100 m respectively. NOTE! IF random MsBsDistance is used, it is important to set feasible values for RMIN and RMAX. This is not done automatically for different scenarios.

5.4 ANSI-C speed-up implementation

The computationally heaviest parts of the WIM channel model have been implemented with ANSI-C, namely the computation of the channel coefficients (‘core’) and interpolation of antenna field patterns. The first one refers to the time-consuming generation of (N paths x M sub-paths) complex exponentials for each user, each one randomly selected for each drop. The latter is based on interpolation functions in GNU Scientific Library (GSL) [5] and can be used only on platforms, where GSL has been installed.

Note! ANSI-C version does not operate with a certain combinations of model options. In this case an error message is given.

5.4.1 A faster channel coefficients generation: scm_mex_core.m

To make use of the optimized computation, one must:

1) Compile the ANSI-C function. The simplest way to do this is to type ‘mex scm_mex_core.c’ at the MATLAB command prompt (provided that MATLAB’s C compiler has been configured properly).
2) Set ‘wimpar.AnsiC_core=’yes’’.

There may be considerable differences between C compilers with respect to the resulting performance.

Further performance improvement may be achieved by setting wimpar. LookUpTable=-1. This activates the lookup table for computing the complex exponential, in the core equation of the channel model. Alternatively, one can set the number of points in the look-up table by e.g. setting wimpar. LookUpTable=1024. The default lookup table size (with LookUpTable = -1) is 2^14=16384.

5.4.2 A faster antenna pattern interpolation function: interp_gain_c.m

In some applications, particularly when the WIM function is called repeatedly for a small number of time samples, antenna field pattern interpolation may constitute a large part of computation. For such applications it may be worthwhile to use the ANSI-C written interpolation function interp_gain_c. The function is based on the interpolation functions in GNU Scientific Library (GSL) and supports linear and cubic spline interpolation with periodic boundary conditions. Look-up table -based interpolation is also supported for uniformly sampled field patterns. To compile the function, type

```
mex -lgsl -lgslcblas -lm interp_gain_mex.c
```

at the MATLAB command prompt. The GNU Scientific Library [5] must be installed in the system for successful compilation. For list of platforms supported by GSL, see [5].

6. References

[1] IST-WINNER II, D1.1.2 "WINNER II Channel Models", ver 1.0, Sep 2007, https://www.ist-winner.org/WINNER2-Deliverables/.
[2] D5.4, "Final Report on Link Level and System Level Channel Models", ver 1.4 , November 2005.
[3] D5.3, "Interim Channel Models", April 2005.
[4] "Spatial channel model for Multiple Input Multiple Output (MIMO) simulations" 3GPP TR 25.996 V6.1.0
[5] GSL – GNU Scientific Library, http://www.gnu.org/software/gsl/
[6] IST-WINNER II, D1.1.1 "WINNER II Interim Channel Models", ver 1.2, Feb 2007, https://www.ist-winner.org/WINNER2-Deliverables/.
[7] EDITORIAL COMMENTS ON CHANNEL MODELS OF ITU-R M.[IMT.EVAL], Contribution 5D/???-E, to be presented in WP5D#3 in Seoul, October 2008.

Appendix A: Terminology and notation

'Link' = one-directional (downlink or uplink) BS-MS connection. The term 'link' is sometimes, but not always, interchangeable with 'user', i.e. MS.

'Channel realisation' = a sequence of channel matrices over a pre-defined number of time samples (in WIM, each element of the channel matrix has six channel clusters in delay domain).

Cluster ≡ tap ≡ path

Ray ≡ subpath

K number of links

N number of paths

M number of subpaths within a path

U number of receiver elements

S number of transmitter elements

T number of time samples